

## README for CSE 464

### Project Part 2

Parth Shah

1225457038

GitHub Repository link: <https://github.com/Parth576/CSE-464-2023-prshah11>

#### 1. Adding Maven support to the project

- I was following the standard directory layout from when I had started my project by following the guide at <https://maven.apache.org/guides/introduction/introduction-to-the-standard-directory-layout.html>
- I created a pom.xml and added all my project dependencies. The commit which contains this change: <https://github.com/Parth576/CSE-464-2023-prshah11/commit/10300a57f930b1adc58d8c1d556a570c4f81b5b4>
- A later commit fixed the tests not running while executing the mvn package command in which the version for maven-surefire-plugin needed to be changed to 2.22.0: <https://github.com/Parth576/CSE-464-2023-prshah11/commit/50798fb2fe3c5b7b79b20df3f6ce5a2653874430>
- Output for mvn package command

```
Run: m: GraphManager [package] >
✓ GraphManager [package]: At 3/28/ 1sec, 956 ms /home/parth/.jdks/openjdk-19.0.2/bin/java -Dmaven.multiModuleProjectDirectory=/home/parth/IdeaProjects...
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.parth.project.GraphManager >-----
[INFO] Building GraphManager 1.0-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ GraphManager ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ GraphManager ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ GraphManager ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/parth/IdeaProjects/GraphManager/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ GraphManager ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.22.0:test (default-test) @ GraphManager ---
[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.parth.project.GraphManagerTest
Graph successfully parsed!
The number of nodes are: 4
The node Labels are:
a
b
c
d
The number of edges are: 4
The nodes with the direction of edges:
a -> b
b -> c
c -> d
d -> a
```

The screenshot displays an IDE interface with a dark theme. The top toolbar includes buttons for Run, Commit, Pull Requests, and various development tools. The main editor area shows the output of a Maven build for a project named 'GraphManager'. The output is as follows:

```
Run: m GraphManager [package] -  
✓ GraphManager [package]: At 3/28/ 1 sec, 956 ms  
The node labels are:  
a  
b  
c  
d  
The number of edges are: 3  
The nodes with the direction of edges:  
a -> b  
c -> d  
d -> a  
  
Graph successfully parsed!  
The number of nodes are: 3  
The node labels are:  
b  
c  
d  
The number of edges are: 2  
The nodes with the direction of edges:  
b -> c  
c -> d  
  
[INFO] Tests run: 12, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.113 s - in com.parth.project.GraphManagerTest  
[INFO]  
[INFO] Results:  
[INFO]  
[INFO] Tests run: 12, Failures: 0, Errors: 0, Skipped: 0  
[INFO]  
[INFO]  
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ GraphManager ---  
[INFO] Building jar: /home/parth/IdeaProjects/GraphManager/target/GraphManager-1.0-SNAPSHOT.jar  
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 1.309 s  
[INFO] Finished at: 2023-03-28T17:01:16-07:00  
[INFO] -----  
  
Process finished with exit code 0
```

The bottom status bar shows the IDE's state: 'Pushed 1 commit to origin/master (20 minutes ago)', 'Run', 'TODO', 'Problems', 'Terminal', 'Services', 'Messages', 'Build', and 'Dependencies'.

## 2. Adding Github CI to the project

- The initial commit where I added the workflow file to the repository is:  
<https://github.com/Parth576/CSE-464-2023-prshah11/commit/6e3dad1f357fd797177a51534986482cde6678e4>
- The workflow runs on every push to the main branch or when a new pull request is opened to the main branch
- The workflow runs the *mvn package* command. Additionally, the build artifacts (JAR file) is stored in the workflow as well, so that we can download the JAR file directly from the workflow. The commit for this is:  
<https://github.com/Parth576/CSE-464-2023-prshah11/commit/e0560a078583d9cf474e641d27ec44beca1d017>
- Some screenshots to show the Github CI working and producing builds of the project

The screenshot shows the GitHub Actions interface for a workflow run. At the top, the repository is identified as 'Parth576 / CSE-464-2023-prshah11' with a 'Private' label. The navigation bar includes links for Code, Issues, Pull requests, Actions (which is highlighted), Projects, Wiki, Security, Insights, and Settings. Below the navigation bar, the workflow is named 'Java CI with Maven'. The specific run is titled 'docs: update README.md with GraphSearch API usage #14' and is marked as successful with a green checkmark. On the left sidebar, under the 'Jobs' section, the 'build' job is selected. The main area displays the 'build' job details, showing it 'succeeded 20 minutes ago in 20s'. A list of steps follows, each with a green checkmark indicating success: 'Set up job', 'Run actions/checkout@v3', 'Set up JDK 19' (which is highlighted), 'Build with Maven', 'Run mkdir artifacts && cp target/\*.jar artifacts', 'Run actions/upload-artifact@v3', 'Post Set up JDK 19', 'Post Run actions/checkout@v3', and 'Complete job'.

Jobs

build

Run details

Usage

Workflow file

Build with Maven

113 The number of edges are: 3  
114 The nodes with the direction of edges:  
115 a -> b  
116 c -> d  
117 d -> a  
118  
119 Graph successfully parsed!  
120 The number of nodes are: 3  
121 The node labels are:  
122 b  
123 c  
124 d  
125 The number of edges are: 2  
126 The nodes with the direction of edges:  
127 b -> c  
128 c -> d  
129  
130 [INFO] Tests run: 12, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.233 s - in com.parth.project.GraphManagerTest  
131 [INFO]  
132 [INFO] Results:  
133 [INFO]  
134 [INFO] Tests run: 12, Failures: 0, Errors: 0, Skipped: 0  
135 [INFO]  
136 [INFO]  
137 [INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ GraphManager ---  
138 [INFO] Building jar: /home/runner/work/CSE-464-2023-prshah11/CSE-464-2023-prshah11/target/GraphManager-1.0-SNAPSHOT.jar  
139 [INFO] -----  
140 [INFO] BUILD SUCCESS  
141 [INFO] -----  
142 [INFO] Total time: 3.426 s  
143 [INFO] Finished at: 2023-03-28T23:53:59Z  
144 [INFO] -----  
  
> Run mkdir artifacts && cp target/\*.jar artifacts  
  
> Run actions/upload-artifact@v3  
  
> Post Set up JDK 19  
  
> Post Run actions/checkout@v3  
  
> Complete job

Parth576 / CSE-464-2023-prshah11 (Private)

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

← Java CI with Maven

docs: update README.md with GraphSearch API usage #14

Summary

Jobs

build

Run details

Usage

Workflow file

Triggered via push 19 minutes ago

Parth576 pushed b5a2f93 master

Status

Success

Total duration

29s

Billable time

1m

Artifacts

1

maven-workflow.yml

on: push

build 20s

Artifacts

Produced during runtime

Name	Size
Package	8.63 KB

### 3. Adding BFS algorithm

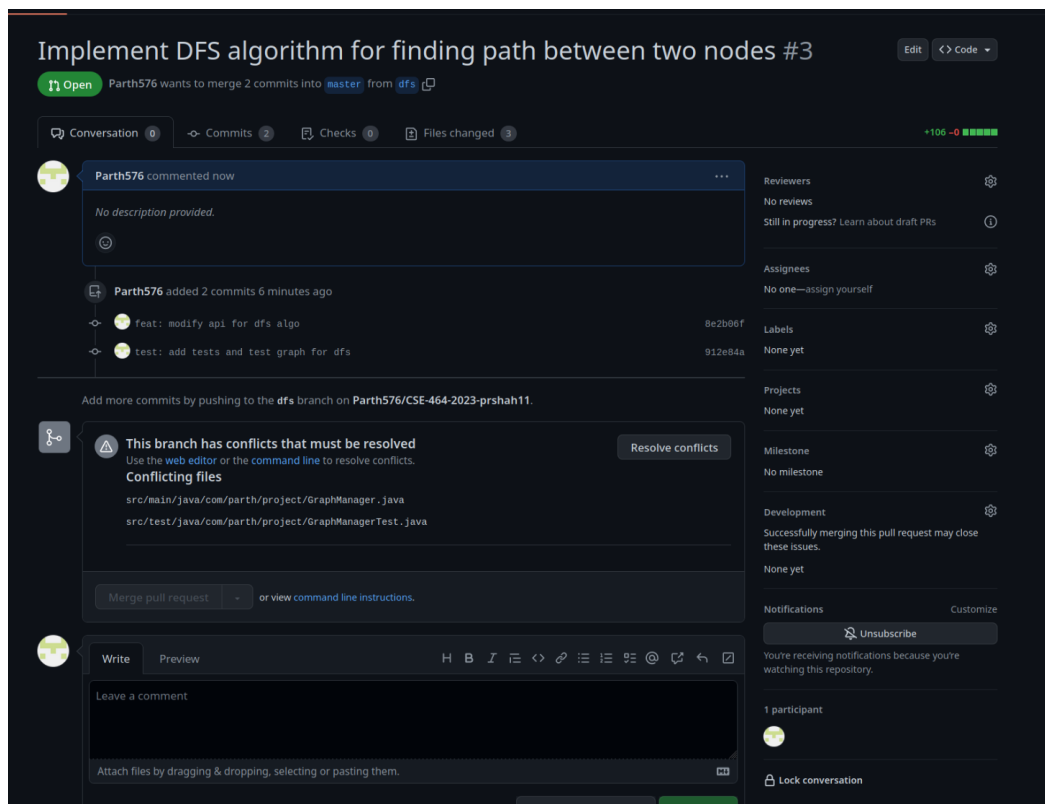
- I created a new API GraphSearch which returns an object of the class Path. It uses a BreadthFirstIterator from jgrapht.
- I created a separate branch 'bfs'  
<https://github.com/Parth576/CSE-464-2023-prshah11/tree/bfs>
- And then opened a pull request to merge the code from the bfs branch to master branch <https://github.com/Parth576/CSE-464-2023-prshah11/pull/1>
- The commit that added the BFS functionality is  
<https://github.com/Parth576/CSE-464-2023-prshah11/pull/1/commits/9e3233078cd260f9da9acc8b34dfc2185123ba46>
- I also added tests for this API in this commit :  
<https://github.com/Parth576/CSE-464-2023-prshah11/pull/1/commits/aa31c569c8fffd49d3b9c410b6abee32187b27d1>

### 4. Adding DFS Algorithm

- I created a separate dfs branch from master which did the same changes but used a DepthFirstIterator  
<https://github.com/Parth576/CSE-464-2023-prshah11/tree/dfs>
- The pull request to merge this :  
<https://github.com/Parth576/CSE-464-2023-prshah11/pull/3>
- Commit that added DFS functionality:  
<https://github.com/Parth576/CSE-464-2023-prshah11/pull/3/commits/8e2b06fcd1f908658bd90a1f0ee53ba531f14059>

## 5. Merging to master and resolving merge conflicts

- I first merged the bfs branch to master
- Then we can see in this screenshot that I got a merge conflict on the pull request which merges dfs branch to master



- I created an enum Algorithm and fixed the tests to resolve the merge conflict. The commit for that is :  
<https://github.com/Parth576/CSE-464-2023-prshah11/commit/3d790c6a8f07a89681feca527ffe748a48a21bc2>

Showing 2 changed files with 41 additions and 11 deletions.

Filter changed files

src

main/java/com/parth/project

GraphManager.java

test/java/com/parth/project

GraphManagerTest.java

```
10 10 import org.jgrapht.nio.dot.DotExporter;
11 11 import org.jgrapht.nio.dot.DotImporter;
12 12 import org.jgrapht.traverse.DepthFirstIterator;
13 + import org.jgrapht.traverse.BreadthFirstIterator;
13 14
14 15 import javax.imageio.ImageIO;
15 16 import java.awt.*;
@@ -26,6 +27,11 @@
26 27
27 28 public class GraphManager {
28 29
30 + enum Algorithm {
31 +     BFS,
32 +     DFS
33 + }
34 +
29 35 public class Path {
30 36     ArrayList<String> nodes;
31 37
@@ -245,11 +251,19 @@ public void outputGraphics(String filePath) throws Exception {
245 251 }
246 252 }
247 253
248 - public Path GraphSearch(String src, String dst) {
254 + public Path GraphSearch(String src, String dst, Algorithm algo) {
249 255 + if (!graph.containsVertex(src) || !graph.containsVertex(dst)) {
250 256     return null;
251 257 }
252 - Iterator<String> iterator = new DepthFirstIterator<>(graph, src);
258 + Iterator<String> iterator;
259 + if (algo == Algorithm.BFS) {
260 +     iterator = new BreadthFirstIterator<>(graph, src);
261 + } else if (algo == Algorithm.DFS) {
262 +     iterator = new DepthFirstIterator<>(graph, src);
263 + } else {
264 +     return null;
265 + }
266 +
253 267 Path path = new Path();
254 268 while(iterator.hasNext()) {
255 269     String node = iterator.next();
@@ -265,8 +279,4 @@ public Path GraphSearch(String src, String dst) {
265 279 }
266 280 }
267 281
```

## 6. Instructions to run BFS and DFS ([LINK](#))

### Example Code

- Creating a new GraphManager object

```
GraphManager g = new GraphManager();
```

- Parsing the graph and printing information

```
g.parseGraph("src/test.dot");  
System.out.println(g.toString());  
g.outputGraph("src/graphinfo.txt");
```

- Find a path from one node to another using BFS or DFS algorithm

```
Path bfs = g.GraphSearch("a", "c", Algorithm.BFS);  
Path dfs = g.GraphSearch("c", "d", Algorithm.DFS);
```

- The `toString()` methods of the `Path` class will print the path in the format `a -> b -> c`
- If no path is found, the `GraphSearch` API returns `null`, so we need to check if the returned path is not `null`
- The `Path` class also exposes the variable `nodes` which is an `ArrayList` containing the searched nodes in the order that they were visited

```
if (bfs != null) System.out.println(bfs.toString());  
if (dfs != null) System.out.println(dfs.toString());  
bfs.nodes;  
dfs.nodes;
```



- There is an example in the test case for that as well

```
@Test
public void testBFS() throws Exception {
    GraphManager gm = new GraphManager();
    gm.parseGraph("src/test2.dot");
    ArrayList<String> expected = new ArrayList<>();
    expected.add("a");
    expected.add("d");
    expected.add("c");
    expected.add("b");
    expected.add("h");
    String expectedString = "a -> d -> c -> b -> h";

    GraphManager.Path result = gm.GraphSearch("a", "h", GraphManager.Algorithm.BFS);

    assertNotNull(result);
    assertEquals(expected, result.nodes);
    assertEquals(expectedString, result.toString());

    result = gm.GraphSearch("h", "a", GraphManager.Algorithm.BFS);
    assertNull(result);
}

@Test
public void testDFS() throws Exception {
    GraphManager gm = new GraphManager();
    gm.parseGraph("src/test2.dot");
    ArrayList<String> expected = new ArrayList<>();
    expected.add("a");
    expected.add("b");
    expected.add("f");
    expected.add("e");
    expected.add("c");
    expected.add("g");
    expected.add("d");
    expected.add("i");
    expected.add("h");
    String expectedString = "a -> b -> f -> e -> c -> g -> d -> i -> h";

    GraphManager.Path result = gm.GraphSearch("a", "h", GraphManager.Algorithm.DFS);

    assertNotNull(result);
    assertEquals(expected, result.nodes);
    assertEquals(expectedString, result.toString());

    result = gm.GraphSearch("h", "a", GraphManager.Algorithm.DFS);
    assertNull(result);
}
```